AMENDMENTS TO THE CLAIMS

Please amend Claims 1, 13, 15, and 21 as follows, without prejudice or disclaimer to continued examination on the merits:

1. (Currently Amended): A computer system, comprising:

a plurality of hardware resources;

a plurality of logical resources; wherein the logical resources are created in a logical model and wherein the logical resources represent hardware resources, wherein the logical model comprises a hardware model that models all hardware available on the computer system, a software model that models each software process available on the computer system, and a service endpoint model that spans the hardware and software models;

a plurality of functional processes, wherein the logical model is used as input to a code generator to create a view and an application programming interface for each of the functional processes that require configuration data;

an operating system that includes memory management which supports a protected memory model, wherein a process is assigned a unique or separate protected memory block, where the processes are decoupled from the system through views of the logical model, said views defining a particular set of data to which an associated process has access, wherein views allow multiple different processes to use the same logical model to be started, upgraded, or restarted independently of other processes;

a configuration process for configuring certain of the plurality of functional processes on particular ones of the logical resources; and

a mapping process for creating a map associating the plurality of hardware resources with the plurality of logical resources; [[and]]

wherein the logical model is layered on the hardware resources, adding a layer of abstraction between the hardware resources and the functional processes;

wherein the functional processes are decoupled from the logical model and the hardware resources with a second layer of abstraction;

wherein the logical model and the hardware resources are changed, evolved, and grown to support new functional processes without having to change existing functional processes; and

wherein software applications may be upgraded and downgraded independent of each other and without having to reboot the computer system.

2. (Original): The computer system of claim 1, wherein the computer system is a network device and wherein the mapping process is a network management system process.

3. (Original): The computer system of claim 1, wherein the map comprises a logical to physical card table.

4. (Original): The computer system of claim 1, wherein the map comprises a logical to physical port table.

5. (Canceled)

6. (Previously Presented): The computer system of claim 1, wherein computer system is a network device and the configuration process is a network management system process.

7. (Canceled)

8. (Original): The computer system of claim 1, wherein the hardware resources include line cards.

9. (Original): The computer system of claim 1, wherein the hardware resources include line card ports.

10. (Previously Presented): The computer system of claim 1, wherein the functional processes include device driver processes.

11. (Previously Presented): The computer system of claim 1, wherein the functional processes include network protocol applications.

12 (Original): The computer system of claim 11, wherein the network protocol applications include Asynchronous Transfer Mode protocol applications.

13. (Currently Amended): A method of operating a computer system, comprising:

providing a modular software architecture, wherein said software architecture comprises a module that resides in a protected memory space, wherein the module has a logical name that can be resolved into a location and a process, wherein during a switchover to a backup module for an upgrade or a failure, the backup module assumes the resolvable logical name, such that computing processes continue unaffected by the switchover;

providing an operating system that includes memory management which supports a protected memory model, wherein a process is assigned a unique or separate protected memory block, such that processes, where said processes are decoupled such that they are decoupled from the system through views of a logical model, said views defining a particular set of data to which an associated process has access, wherein views allow multiple different processes to use the same logical model to be started, upgraded, or restarted independently of other processes;

wherein the logical model comprises a hardware model that models all hardware available on the computer system, a software model that models each software process available on the computer system, and a service endpoint model that spans the hardware and software models;

wherein the logical model is used as input to a code generator to create a view and an application programming interface for each of the processes that require configuration data;

creating logical resources having characteristics similar to particular hardware resources;

generating a map of logical resources to hardware resources; and

provisioning services to logical resources; [[and]]

wherein the logical model is layered on the hardware available on the computer system, adding a layer of abstraction between the hardware available on the computer system and the functional processes;

wherein the processes are decoupled from the logical model and the hardware available on the computer system with a second layer of abstraction;

wherein the logical model and the hardware available on the computer system are changed, evolved, and grown to support new processes without having to change existing

processes; and

wherein software applications may be upgraded and downgraded independent of each other and without having to reboot the computer system.

14. (Original): The method of claim 13, wherein the computer system is a network device and wherein generating a map is accomplished through the use of a network management system process.

15. (Currently Amended): The method of claim 13, wherein the map comprises a logical to physical ~~port~~ card table.

16. (Original): The method of claim 13, wherein the map comprises a logical to physical port table.

17. (Canceled)

18. (Canceled)

19. (Canceled)

20. (Original): The method of claim 13, wherein the hardware resources include line card.

21. (Currently Amended): The method of operating a computer system, comprising:

providing a modular software architecture, wherein said software architecture comprises a module that resides in a protected memory space, wherein the module has a logical name that can be resolved into a location and a process, wherein during a switchover to a backup module for an upgrade or a failure, the backup module assumes the resolvable logical name, such that computing processes continue unaffected by the switchover;

providing an operating system that includes memory management which supports a protected memory model, wherein a process is assigned a unique or separate protected memory block, where said processes are decoupled such that they are decoupled from the

system through views of a logical model, said views defining a particular set of data to which an associated process has access, wherein views allow multiple different processes to use the same logical model to be started, upgraded, or restarted independently of other processes;

wherein the logical model comprises a hardware model that models all hardware available on the computer system, a software model that models each software process available on the computer system, and a service endpoint model that spans the hardware and software models;

wherein the logical model is used as input to a code generator to create a view and an application programming interface for each of the processes that require configuration data;

providing a logical resource;

providing a physical resource;

configuring the process on the logical resource, said logical resource providing a model of the physical resource; and

applying the configured logical resource to the physical resource; [[and]]

wherein the logical model is layered on the hardware available on the computer system, adding a layer of abstraction between the hardware available on the computer system and the functional processes;

wherein the processes are decoupled from the logical model and the hardware available on the computer system with a second layer of abstraction;

wherein the logical model and the hardware available on the computer system are changed, evolved, and grown to support new processes without having to change existing processes; and

wherein software applications may be upgraded and downgraded independent of each other and without having to reboot the computer system.

22. (Previously Presented): The method of claim 21, further comprising:

applying the configured logical resource to another physical resource.

23. (Previously Presented): The method of claim 21, further comprising:

detecting a fault on the physical resource;

failing over from the physical resource to a second physical resource; and

applying the configured logical resource to the second physical resource.

24. (Previously Presented): The method of claim 21, further comprising:

detecting an event within the computer system; and

applying the configured logical resource to a second physical resource.

25. (Previously Presented): The method of claim 24, wherein the event includes a resource consumption notification.

26. (Previously Presented): The method of claim 24, wherein the event includes a fault.

27. (Previously Presented): The method of claim 21, wherein the process comprises a first process, the logical resource comprises a first logical resource and the physical resource comprises a first physical resource and further comprising:

configuring a second process on a second logical resource; and

applying the configured second logical resource to a second physical resource.

28 (Previously Presented): The method of claim 27, wherein the first and second processes are the same process.

29. (Previously Presented): The method of claim 27, wherein the first and second processes are different processes.

30. (Previously Presented): The method of claim 27, wherein the first and second logical resources are the same logical resource.

31. (Previously Presented): The method of claim 27, wherein the first and second logical resources are different logical resources.

32. (Previously Presented): The method of claim 27, wherein the first and second physical resources are the same hardware resource.

33. (Previously Presented): The method of claim 27, wherein the first and second hardware resources are different hardware resources.

34. (Previously Presented): The method of claim 21, wherein configuring the process on the logical resource comprises:

filling in a field in a table in a configuration database.

35. (Previously Presented): The method of claim 21, wherein configuring a process on the logical resource comprises:

filling in a plurality of fields in a plurality of tables in a configuration database.

36. (Previously Presented): The method of claim 35, wherein the plurality of tables comprise an application group table.

37. (Previously Presented): The method of claim 35, wherein the plurality of tables comprise an application interface table.

38. (Previously Presented): The method of claim 35, wherein the plurality of tables comprise a service end point table.

39. (Previously Presented): The method of claim 21, wherein applying the configured logical resource to the physical resource comprises:

assigning a logical identifier to the physical resource.

40. (Previously Presented): The method of claim 39, wherein applying the configured logical resource to the physical resource further comprises:

filling in a field in a table in a configuration database.

41. (Previously Presented): The method of claim 40, wherein the table comprises a logical to physical card table.

42. (Previously Presented): The method of claim 21, wherein the logical resource represents a physical hardware module and the physical resource comprises the physical hardware module.

43. (Previously Presented): The method of claim 42, wherein the physical hardware module comprises a card.

44. (Previously Presented): The method of claim 43, wherein the card comprises a line card.

45. (Previously Presented): The method of claim 43, wherein the card comprises a physical card.

46. (Previously Presented): The method of claim 21, wherein the physical hardware module comprises a board.

47. (Previously Presented): The method of claim 46, wherein the board comprises a central processing board.

48. (Previously Presented): The method of claim 21, wherein the logical resource represents a physical port on a forwarding card and the physical resource comprises the physical port on the forwarding card.

49. (Previously Presented): The method of claim 48, wherein the logical resource comprises a service endpoint and the physical port comprises a port on a forwarding card.

50. (Previously Presented): The method of claim 21, wherein the logical resource comprises a logical identifier.

51. (Previously Presented): The method of claim 21, wherein the computer system comprises a network device.

52. (Previously Presented): The method of claim 51, wherein configuring the process on the logical resource comprises:

configuring network connectivity on the logical resource;.

53. (Previously Presented): The method of claim 21, further comprising:

adding the physical resource to the computer system, wherein applying the configured logical resource to the physical resource is delayed until the physical resource is added to the computer system.

54. (Previously Presented): The method of claim 21, wherein configuring the process on the logical resource comprises:

configuring a plurality of processes on a plurality of logical resources; and wherein applying the configured logical resource to the physical resource comprises:

applying the configured plurality of logical resources to a plurality of physical resources.

55. (Previously Presented): The method of claim 21, wherein the process comprises an application.